Automated Testing of Image Captioning Systems

Boxi Yu
221049024@link.cuhk.edu.cn
The Chinese University of Hong Kong, Shenzhen
Shenzhen, Guangdong, China

Zhiqing Zhong
cheehingchung@gmail.com
South China University of Technology
Guangzhou, Guangdong, China

Xinran Qin
csqinxinran@mail.scut.edu.cn
South China University of Technology
Guangzhou, Guangdong, China

Jiayi Yao
120040070@link.cuhk.edu.cn
The Chinese University of Hong Kong, Shenzhen
Shenzhen, Guangdong, China

Yuancheng Wang
119010319@link.cuhk.edu.cn
The Chinese University of Hong Kong, Shenzhen
Shenzhen, Guangdong, China

Pinjia He*
hepinjia@cuhk.edu.cn
The Chinese University of Hong Kong, Shenzhen
Shenzhen, Guangdong, China

ABSTRACT

Image captioning (IC) systems, which automatically generate a text description of the salient objects in an image (real or synthetic), have seen great progress over the past few years due to the development of deep neural networks. IC plays an indispensable role in human society, for example, labeling massive photos for scientific studies and assisting visually-impaired people in perceiving the world. However, even the top-notch IC systems, such as Microsoft Azure Cognitive Services and IBM Image Caption Generator, may return incorrect results, leading to the omission of important objects, deep misunderstanding, and threats to personal safety.

To address this problem, we propose MetaIC, the *first* metamorphic testing approach to validate IC systems. Our core idea is that the object names should exhibit directional changes after object insertion. Specifically, MetaIC (1) extracts objects from existing images to construct an object corpus; (2) inserts an object into an image via novel object resizing and location tuning algorithms; and (3) reports image pairs whose captions do not exhibit differences in an expected way. In our evaluation, we use MetaIC to test one widely-adopted image captioning API and five state-of-the-art (SOTA) image captioning models. Using 1,000 seeds, MetaIC successfully reports 16,825 erroneous issues with high precision (84.9%-98.4%). There are three kinds of errors: misclassification, omission, and incorrect quantity. We visualize the errors reported by MetaIC, which shows that flexible overlapping setting facilitates IC testing by increasing and diversifying the reported errors. In addition, MetaIC can be further generalized to detect label errors in the training dataset, which has successfully detected 151 incorrect labels in MS COCO Caption, a standard dataset in image captioning.

*Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ISSTA '22, July 18–22, 2022, Virtual, South Korea
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9379-9/22/07…$15.00
https://doi.org/10.1145/3533767.3534389

CCS CONCEPTS

• **Software and its engineering → Software testing and debugging**.

KEYWORDS

Metamorphic testing, testing, image captioning, AI software

ACM Reference Format:
Boxi Yu, Zhiqing Zhong, Xinran Qin, Jiayi Yao, Yuancheng Wang, and Pinjia He. 2022. Automated Testing of Image Captioning Systems. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '22), July 18–22, 2022, Virtual, South Korea.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3533767.3534389

1 INTRODUCTION

Image captioning (IC) systems aim to automatically generate a brief depiction of the salient objects in an image (real or synthetic). In recent years, the performance of IC systems [37, 41, 47, 78, 85, 90] have improved significantly due to the rapid development of the underlying deep neural networks, such as convolutional neural networks (CNN) [32, 63] for image feature extraction and language models [24, 36] for caption generation based on these features. The SOTA IC models have reached or even surpassed human-level performance (*e.g.*, VIVO [37]) in terms of CIDEr score [77]. Thus, IC systems have been increasingly integrated into our daily lives. Typical examples include generating captions for massive social media photos [3], visually-impaired people's artificial intelligence (AI) assistants [2, 58], and the acceleration of various manual image labeling tasks [4]. Since 2016, the major IT companies (*e.g.*, Google [4], Microsoft [5], and IBM [1]) have released and continuously improved their own IC systems. The leading company of Geographic Information System (GIS), Ersi, has integrated image captioning into its famous GIS framework (*e.g.*, Arcgis[1]) to generate the captions for remote sensing images.

Despite its wide adoption in various applications, modern IC system could return incorrect captions due to the complexity of deep neural networks and the labeling errors in training datasets, leading to the omission of important objects, deep misunderstanding, and even threats to personal safety [58, 59, 66]. According to the "report on vision" [60], more than 2.2 billion people around the world have vision impairment and they are expected to benefit

[1]https://developers.arcgis.com/python/guide/how-image-captioning-works/

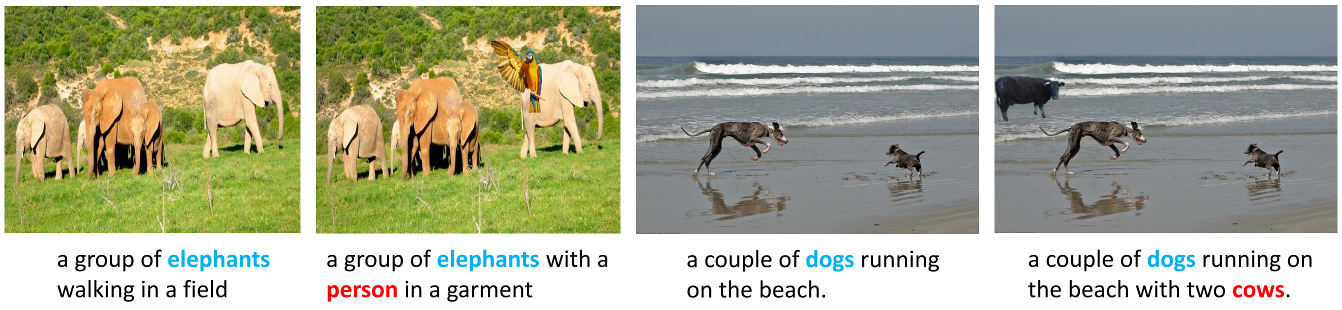

a group of elephants walking in a field
a group of elephants with a person in a garment
a couple of dogs running on the beach.
a couple of dogs running on the beach with two cows.

Figure 1: Examples of background images, generated images, and the corresponding captions. Words of salient objects are marked in blue. Words indicating the violation of MetaIC's MRs are marked in red.

greatly from AI-assistants in which IC is often a major component [2, 58]. However, the embedded IC systems sometimes fail to generate the correct captions [59], which make the AI-assistants return misleading message to the visually-impaired users, posing threat to their personal safety. An image of the former U.S. president Barach Obama and his wife was captioned "a man in suit and tie talking on a cell phone" [66], leading to misunderstanding and potential negative social impact. Thus, assuring the reliability of IC systems is an important endeavor.

There remains a dearth of automated testing methods for IC systems because the problem is quite challenging. First, modern IC systems are approaching human-level performance on the existing human-labeled test sets. Thus, test sets of high quality are lacking. Second, traditional code-based testing approaches [15, 64, 79] are not suitable for testing IC systems because the logics of IC systems are mainly encoded in the underlying neural networks with millions of complex parameters, rather the source code. Third, existing testing approaches for computer vision (CV) tasks [74, 82] typically assume simpler output formats (*e.g.*, class labels), which cannot work well in testing IC systems whose output (*i.e.*, a sentence) is significantly more complex; while existing testing approaches for natural language processing (NLP) [33, 34, 70] tasks heavily rely on the perturbation of chars or words in the input, which is infeasible for IC systems whose input are images.

To tackle these challenges, we introduce MetaIC, a simple, widely-applicable metamorphic testing methodology for validating IC systems. The input of MetaIC is a list of unlabeled images, while its output is a list of suspicious issues, where each suspicious issue contains a pair of images and their captions. The core idea of MetaIC is that the object names should exhibit directional changes after object insertion. We realize this idea by two metamorphic relations (MRs): (1) *object appearance*: only the original objects and the inserted objects should appear in the caption of the generated image; (2) *singular-plural form*: both the original objects and the inserted objects should be illustrated by words of a proper singular-plural form. Fig. 1 presents two suspicious issues returned by our approach. The first issue was reported because the inserted bird was captioned into a "person", violating the first MR; while the second issue was reported because the noun "cow" was not in singular form, violating the second MR. Specifically, MetaIC automatically extracts objects from images by Yolact++ [12], an advanced real-time image segmentation technique. The extracted object will be inserted into randomly selected images (*i.e.*, background images). This is a challenging step because the inserted object should avoid affecting the saliency of the existing objects, which has been tackled by our novel object resizing and location tuning algorithms. The background image and the newly generated image form an image pair. If the captions of this image pair returned by the IC system violate any of the MRs, the image pair and the corresponding captions will be reported as a suspicious issue.

We apply MetaIC to test one paid IC service, *i.e.*, Microsoft Azure Cognitive Services [5] and five popular IC models, *i.e.*, Show, Attend and Tell [85], $Oscar_B$, $Oscar_L$ [47], $VinVL_B$, $VinVL_L$ [90]. MetaIC successfully reports 16,825 erroneous issues in total with high precision of 84.9%-98.4%, revealing 17,380 captioning errors. The errors include misclassification, omission, and incorrect quantity. To better understand the reported errors, we visualize the attention masks of the object, which helps us explore the potential root causes behind. Furthermore, we adapt MetaIC to detect labeling errors in MS COCO Caption, a widely-used standard IC dataset, which reports 151 incorrect labels in the training set, demonstrating MetaIC's wide applicability. The source code is also released for reuse [7]. In summary, this paper makes the following main contributions:

- It introduces MetaIC, the *first* metamorphic testing methodology for validating image captioning systems.
- It describes a practical implementation of MetaIC by adapting Yolact++ [12] to extract objects and developing a new object insertion technique that allows flexible insertion position and overlapping area.
- It presents the evaluation of MetaIC that successfully reports 16,825 erroneous issues in one industrial IC service and five SOTA IC models with high precision, and find a total number of 17,380 captioning errors.
- It discusses the error categories found by MetaIC, the exploration of root causes, and the potential of adapting MetaIC to detect labeling errors in widely-used image captioning datasets.

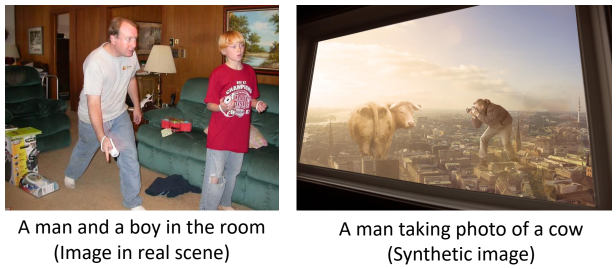


Figure 2: A real image, a synthetic image, and their captions.

2 PRELIMINARIES

In this section, we first explain the basic concepts that have been used multiple times in the paper. Then we introduce two mainstream image captioning models: CNN-RNN-Based IC and Vision-Language Pre-Training IC.

2.1 Basic Concepts

Image captioning outputs a depiction of the *salient objects* in a given image (real or synthetic), where salient objects refer to the most noticeable object(s) in the image; and *saliency* reflects whether an object is salient or not. For example, in Fig. 2, the two people are salient objects in the real-scene image, and the man and the cow are salient objects in the synthetic image. Each object has an *object class*, such as "elephant" and "dog" in Fig. 1. In a caption returned by IC software, a salient object is typically illustrated by a noun, whose *singular-plural form* indicates whether the word is a singular noun or a plural noun.

In this paper, we test one IC service and five IC models: Microsoft Azure API [5], Attention [85], $Oscar_B$, $Oscar_L$ [47], $VinVL_B$, $VinVL_L$ [90]. Specifically, Microsoft Azure is short for Microsoft Azure Cognitive services [5] and Attention is short for Show, Attend and Tell [85]. $Oscar_B$ and $VinVL_B$ refer to the IC models adopting the base version of BERT [24], while $Oscar_L$ and $VinVL_L$ adopt the large version.

2.2 Modern Image Captioning Systems

Recently, there are two main lines of IC research: CNN-RNN-Based IC [78, 85] and Vision-Language Pre-Training (VLP) IC [37, 47, 90, 92]. They both adopt a two-stage framework, using CNNs for feature extraction in the first stage and sequence models caption generation in the second stage. Differently, VLP IC adopts pre-trained models in the second stage, which makes them more efficient and accurate than CNN-RNN-Based IC. In particular, the SOTA VLP IC model VIVO [37] claims that it surpasses human in CIDER score [77]. In the following, we elaborate more on these two lines of research.

2.2.1 *CNN-RNN-Based ICs.*

CNN-RNN-Based IC uses a CNN as the feature extractor and an RNN to generate the captions. Specifically, CNN can produce rich representation of an image by embedding it into a fixed-length vector and the representation has been used in various CV tasks [68]. Inspired by the great successes of sequence generation in machine translation [10, 22, 71], RNN models [41, 78, 85] have been adopted in caption generation in CNN-RNN-Based IC. Attention [85] first introduces an attention mechanism to image captioning and visualize how the model attends on the salient objects in an image during captioning. By using a lower convolutional layer rather than fully connected layers in the encoder, the RNN decoder of this method can focus on the most important parts of the image by calculating weights from the feature vectors. As for the second stage, it adopts LSTM [36] for caption generation.

2.2.2 *Vision-Language Pre-Training ICs.*

VLP ICs also adopts a two-stage pipeline. Differently, they employ more advanced CNNs in the first stage and BERT [24] in the second stage. With a multi-layer bidirectional Transformer [76] as its architecture, BERT uses masked language models to enable pre-trained deep bidirectional representations, which effectively alleviate the extensive manual effort on building task-specific architectures.

Oscar [47] first adopts object tags detected in images as anchor points, which strengthens the learning of semantic alignments between images and texts and significantly enhance its ability to learn the cross-modal representations. Recently, Zhang *et al.* [90] propose VinVL, which uses a CNN specially tailored for vision-language tasks. This CNN is pre-trained on a much larger text-image corpora, thus it can capture much more abundant visual features accurately, outperforming other models.

3 APPROACH AND IMPLEMENTATION

This section introduces MetaIC and its implementation details. Inspired by the goal of IC that the salient objects in an image should be captioned properly, the core idea of MetaIC is: the object names should exhibit directional changes after object insertion. The input of MetaIC is a list of unlabeled images and the output is a list of suspicious issues, where each issue contains the original image, a generated image with an inserted object, and their corresponding captions returned by the IC software under test. Fig. 3 illustrates the overview of MetaIC, which carries out the following four steps:

(1) *Object extraction.* We extract salient objects from an existing dataset by an instance segmentation algorithm, which collectively form an object pool.
(2) *Object insertion.* For each unlabeled image (*i.e.*, background image), we randomly select an image (*i.e.*, object image) from the object pool and insert it into the background image via novel object resizing and location tuning algorithms.
(3) *Caption collection.* We collect the captions for the background image and the generated image from the IC system under test.
(4) *Error detection.* We focus on the key constituents in the two captions and report a suspicious issue if the captions violate at least one of our MRs.

3.1 Object Extraction

To realize object insertion, we need to prepare a pool of object images. Thus, the first step of MetaIC is to extract object images from an image dataset. Although object extraction has long been a challenging task, deep learning-based object segmentation models [12, 13, 31, 48] provide a feasible solution because these models can extract high-quality objects efficiently.

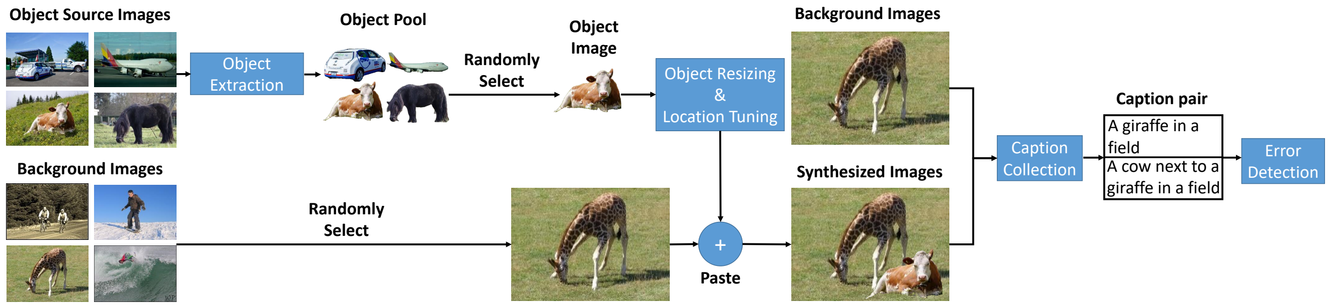


Figure 3: Overview of MetaIC.

In our implementation, we adopt Yolact++ [12], the SOTA real-time object segmentation method because it achieves decent performance in terms of both precision [31, 48] and efficiency [12, 13, 75]. Yolact++ predicts mask prototypes and per-instance mask coefficients in parallel, and linearly combines them to form the final instance masks. Thus, Yolact++ model can achieve 34.1 mAP on MSCOCO [51] at 33.5 fps while keeping the close performance of the SOTA approaches. As shown in Fig. 4, Yolact++ generates the masks and identifiers of the airplanes in the images, based on which we can extract the objects accordingly and put them into a certain category (*e.g.*, airplane).

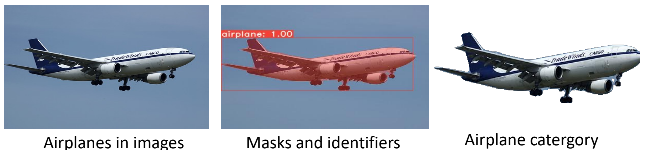


Figure 4: An example of object extraction via Yolact++.

3.2 Object Insertion

Once we have constructed an object pool, we randomly select an object image (*e.g.*, the cow in Fig. 3) from the pool and insert it into a random background image. In this process, we need to address three main issues: (1) how to resize the object image to avoid being too big or too small compared with the salient objects in the background image; (2) how to find a suitable location for insertion; and (3) how to allow reasonable overlap between the object image and the original objects in the background image. In the following, we will introduce our specially-design object resizing and location tuning algorithm and how these algorithms provide us with the flexibility of overlapping area configuration.

3.2.1 Object resizing. The goal of object resizing is to resize the object image from the original size (h, w) to a new size (h', w'), where h and h' refer to their heights and w and w' refer to their widths. MetaIC is based on the assumption that the inserted object is a salient object and it should not affect the saliency of the objects in the background image. Thus, the resized object image should not be too small nor too big. Our object resizing algorithm is illustrated by Fig. 5. To figure out the feasible (h', w'), we first determine the area size $s \approx h' \times w'$ by randomly selecting a value from range $[S_{min}, S_{max}]$, where S_{min} and S_{max} are calculated as follows:

$$\begin{aligned} S_{min} &= \alpha * S(b) \\ S_{max} &= \beta * S(b) \end{aligned} \tag{1}$$

α and β are hyper-parameters that we manually set based on empirical experience; $S(b)$ is a normalized value that reflects the area size of the objects in the background image. Specifically, $S(b)$ is calculated by $Softmax$ function as follows:

$$\begin{aligned} S(b) &= \sum_{i=1}^{N} (Softmax(\frac{a_i}{s_b}) * a_i) \\ Softmax(x_i) &= \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}} \end{aligned} \tag{2}$$

where a_i is the area size of the i-th original objects in the background image b; s_b is the area size of b; N is the number of objects in b. After randomly selecting an area size s from the interval $[S_{min}, S_{max}]$, we calculate (h', w') as follows:

$$(h', w') = (h * \sqrt{\frac{s}{h * w}}, w * \sqrt{\frac{s}{h * w}}) \tag{3}$$

We denote a_{max} as the area size of the largest object in b. Empirically when $\frac{a_{max}}{s_b} < 40\%$, we set α, β as 0.8 and 1.3, respectively; otherwise, we set α, β as 0.1 and 0.37, respectively.

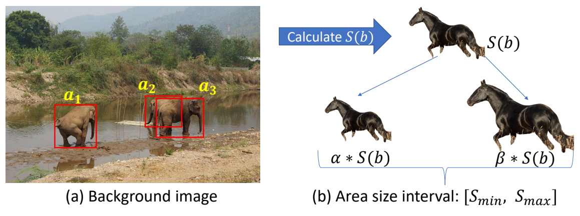


Figure 5: An example of object resizing.

3.2.2 Location tuning. After resizing the object image, we need to select suitable locations in the background image for object insertion. MetaIC provides a flexible setting on the overlapping ratio between the inserted object and objects in the background image. Specifically, MetaIC can generate n images with different overlapping ratios, which are randomly selected from n ratio intervals given a maximum overlapping ratio $ratio_{max}$. The intervals for the overlapping ratios are calculated as follow:

$$interval_i = \begin{cases} [0] & \text{if } i = 0 \\ (\frac{ratio_{max}}{n-1} * (i-1), \frac{ratio_{max}}{n-1} * (i)] & \text{if } i > 0 \end{cases} \quad (4)$$

For example, if we set $ratio_{max} = 0.45$ and $n = 4$, we will have four intervals: *i.e.*, [0], (0, 0.15], (0.15, 0.3], (0.3, 0.45]), where "[0]" indicates that the inserted object should not overlap with any of the objects in the background image. The overlapping ratio between two objects is the overlapping ratio between the bounding boxes of the inserted object and that of the objects in the background image. Specifically, the overlapping ratio O_j is defined as:

$$O_j = \frac{A_j \cap A_{obj}}{A_j}, \quad (5)$$

where A_j is the region of the j-th object in the background image, A_{obj} is the region of the inserted object obj.

For each of the ratio intervals, MetaIC generates one synthetic image by selecting a location (x, y) for object image insertion, leading to overlapping ratio O_j within the interval for all the objects. To ensure the overlapping ratio while keeping the saliency of the original objects, MetaIC requires: (1) O_j should be in the ratio interval if the j-th object is the largest object in the background image (Rule R_1); (2) otherwise, O_j should be less than or equal to the upper bound of the ratio interval Rule R_2.

It is challenging to select suitable positions that obeys rules R_1 and R_2 in an efficient manner because there could be multiple objects in a background image, where each poses an overlapping constraint according to the rules. To tackle this challenge, we develop a two-step algorithm. We first search a suitable coordinate for object insertion that allows an overlapping ratio in $interval_{n-1}$, *e.g.*, (0.3, 0.45] when $ratio_{max} = 0.45$ and $n = 4$. Then we searches the suitable positions for other ratio intervals only along a line instead of all possible positions. Alg. 1 presents pseudo-code.

Step 1: We randomly choose a coordinate (x_{n-1}, y_{n-1}) in the background image for object insertion for $interval_{n-1}$, and check if it obeys R_1 and R_2 (line 3-7). If not, MetaIC calculates a new image size using our image resizing technique, generates a new coordinate (x, y) randomly, and check its compliance with R_1 and R_2 again. We keep generating new image sizes and coordinates until they obey R_1 and R_2 or we try c_1 times, where c_1 is a pre-defined threshold (line 7-11).

Step 2: After calculating the coordinate (*e.g.*, point S in Fig. 6 (a)) for $interval_{n-1}$, we continue to search for the object insertion coordinates under the remaining ratio intervals. Instead of utilizing the strategy in *step-1* to find a new coordinate, which is time-consuming, we search the coordinate along a line. We denote the centroid coordinate of the largest object in background image as (x_{max}, y_{max}), which is point A in Fig. 6 (a) (line 17). We construct a line L (red lines in Fig. 6) by connecting point (x_{max}, y_{max}) and point (x_{n-1}, y_{n-1}) and extend the line to point (x_{bound}, y_{bound}) (point B in Fig. 6 (a)), where (x_{bound}, y_{bound}) is the farthest coordinate on L keeping the inserted object inside the background image. MetaIC then employs binary search to find a suitable coordinate that obeys the rules R_1 and R_2 along the line (line 18). Fig. 6 (b), (c), (d) demonstrate an example of the binary search process along the line when MetaIC intends to find a suitable coordinate for ratio interval (0, 0.15]. In

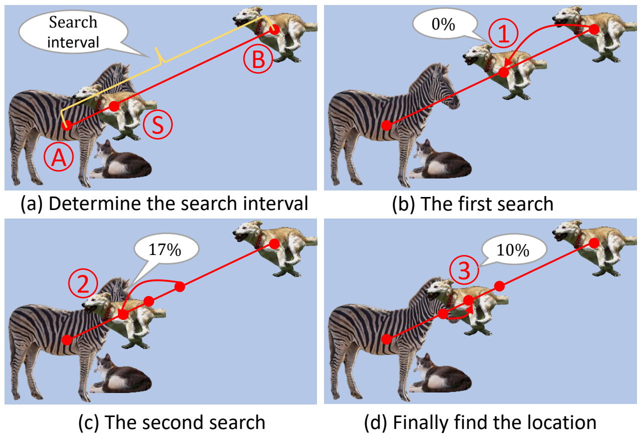


Figure 6: An example of our location technique.

the first search, it gets an overlapping ratio that equals to 0% which is too small. Then it performs the second search and gets the overlapping ratio that equals to 17%, which violates $R1$. Finally, we find coordinate (x_1, y_1) that satisfies R_1 and R_2 for the ratio interval. We could see that the saliency of the cat is not affected by the inserted dog after insertion. For each ratio interval, we allows at most c_2 "jumps" in the binary search, where c_2 is a pre-defined threshold. If we cannot find a suitable position for the interval, we will randomly select a new image and starts from *step-1* again.

Algorithm 1 An implementation of location tuning

Input: the background b, a list of object candidates $objects$, number of overlapping ratios n, the max overlapping ratio $ratio_{max}$, resizing interval parameters α and β, patience parameters c_1 and c_2
Output: a size for the inserted object, and a list of locations for controlling different overlapping ratios

```
1:  $location\_list \leftarrow List()$                         ▷ Initialize with empty list
2:  for $obj$ in $objects$ do
3:      $coordinate_{max} \leftarrow$ RandomCoordinate( )
4:      $ObjNewSize \leftarrow$ RandomSize($\alpha, \beta, obj, b$)
5:      $obj \leftarrow$ Resize($obj, ObjNewSize$)
6:      $patience_1 \leftarrow 0$
7:      while $R_1$ is False or $R_2$ is False for $ratio_{max}$ do     ▷ Step 1
8:          if $patience > c_1$ then
9:              break
10:         $coordinate_{max} \leftarrow$ RandomCoordinate( )
11:         $patience_1 += 1$
12:     for i in range(0,n-2) do                                 ▷ Step 2
13:         $patience_2 \leftarrow 0$
14:         while $R_1$ is False or $R_2$ is False for $ratio_i$ do
15:             if $patience > c_2$ then
16:                 break
17:             $centroid \leftarrow$ Coordinate(LargestObjOf(b))
18:             $coordinate_i \leftarrow$ BinarySearch($centroid$, Boundary(b))
19:             $patience_2 += 1$
20:         $location\_list$.append($coordinate_i$)
21:     if $Length(location\_list) == n$ then
22:         break
23: return $ObjNewSize, location\_list$
```

3.3 Caption Collection

After object insertion, MetaIC constructs image pairs, where each pair contains the background image and a corresponding synthetic image. These images will be input to the IC system under test and the returned captions will be collected. In this paper, we test one paid IC service, *i.e.*, Microsoft Azure Cognitive Services [5], and five popular IC models, *i.e.*, Attention [85], $Oscar_B$, $Oscar_L$ [47], $VinVL_B$, $VinVL_L$ [90]. For the paid IC service, we invoke the API provided by Microsoft Azure Cognitive Services [5]. For the IC models, we use the open-source code provided by the authors and train our own IC models following the exact description in the original papers.

3.4 Error Detection

After caption collection, MetaIC inspects the captions of every image pair and reports them as a suspicious issue if they violate our metamorphic relations (MRs). MetaIC provides two MRs.

MR1. We denote the IC system as I, the background image as b, the generated image as i', the inserted object as obj, the caption pair produced by I are $I(b)$ and $I(i')$. *MR1* is defined as follows:

$$Set(I(i')) == Set(I(b)) \cup \{obj\}, \tag{6}$$

where $Set(I(i'))$ denotes the set of object classes in $I(i')$, $Set(I(b))$ indicates the set of object classes in $I(b)$.

MR2. We use a function $F_*(y)$ to denote the singular-plural form of an object class y in caption $I(*)$, indicating whether the object class y's form is singular or plural. For example, $F_b(dog)$ is the singular-plural form of "dog" class in the caption of background image b. C denotes the object classes in both $I(b)$ and $I(i')$; c_k is the k-th object class in C. If the inserted object obj is not in $Set(I(b))$, *MR2* requires:

$$\begin{aligned} F_b(c_k) &= F_{i'}(c_k), \\ F_{i'}(obj) &= singular. \end{aligned} \tag{7}$$

If the inserted object obj is in $Set(I(b))$, we consider $C' = C \setminus \{obj\}$, where c_j is the j-th object class in C'. Then the second MR requires:

$$\begin{aligned} F_b(c_j) &= F_{i'}(c_j), \\ F_{i'}(obj) &= plural. \end{aligned} \tag{8}$$

If any of the two MRs are violated, the original image, the generated image, and the corresponding captions will be reported as a suspicious issue.

In our implementation, we mark up the word in caption according to a particular part of speech (POS) via a POS tagging tool, *i.e.*, XPOS [56]. We extract words from the captions whose POS are NN or NNS, corresponding to singular noun or plural noun, respectively. Then we can obtain the a set containing all the object classes in a caption, and construct the mapping for function F which assigns the singular-plural form to the object class in the set.

4 EVALUATION

4.1 Experimental Setup and Dataset

4.1.1 Experimental environments. All experiments are run on a Linux workstation with an 8 core AMD Ryzen 5800X 4.8GHz Processor, 64GB DDR4 2666MHz Memory, and GeForce RTX 3090 GPU. The Linux workstation is running 64-bit Ubuntu 20.04.2 LTS with Linux Kernel 5.11.0. For POS tagging, we use the XPOS implemented in Stanza,[2] an NLP Package powered by Stanford NLP Group.

4.1.2 Dataset. To show that MetaIC can work effectively on different image sources, we collect object source images from Flickr [6] and background images from MS COCO Caption [21], a standard dataset in image captioning field. For MS COCO Caption Dataset, we focus on the images whose class names are single words (60 out of 80) because the POS tagging technique can only assign POS to single words. It only requires decent engineering effort to generalize MetaIC to class names containing multiple words. All the code and datasets in this paper will be open-source.

4.2 Precision

MetaIC automatically reports suspicious issues, where each issue contains a pair of images and their captions $(I(b), I(i'))$. Therefore, the effectiveness lies in how precise the reported issues are. In this section, we try to answer the following question: how many of the reported issues contain real captioning errors. We test one well-known IC API: Microsoft Azure API, and five IC models: Attention [85], $Oscar_B$, $Oscar_L$ [47], $VinVL_B$, $VinVL_L$ [90]. In this experiment, we set $ratio_{max} = 45\%$ and $n = 4$ to balance the diversity of the test cases (in terms of the overlapping ratio) and the test case quality. We believe this parameter setting can be used in other datasets in general because the MS COCO Caption contains diverse images (different sizes and salient objects). We use MetaIC to randomly generate 1,000 background-object pairs from MS COCO Caption and our object corpus, and construct 1,000 synthesized images for every overlapping ratio interval. After synthesizing the images, we obtain 4,000 image pairs (b, i'). We collect 4,000 caption pairs $(I(b), I(i'))$ from each of the IC systems under test. Based on these caption pairs, MetaIC returns a list of suspicious issues.

We are the first to test IC systems so there are no available baselines. Intuitively, the metamorphic testing techniques for simpler tasks that also take images as input could be adapted to testing IC systems. Thus, we adopt Deeptest [73], a metamorphic testing technique for image classifier in our precision experiment. Specifically, we use four image transformations in Deeptest (blur, brightness, contrast, shear) and check whether the image pair share the same caption. We use the same background images as the experiment setting of MetaIC to synthesize 4,000 image pairs (b, i'), and collect 4,000 caption pairs $(I(b), I(i'))$ from the IC systems.

To verify the results, two authors manually inspect all the suspicious issues separately following the instructions on data labeling in MS COCO paper [21], including "Describe all the important parts of the scene", "Do not describe things that might have happened in the future or past", *etc.* During this manual analysis, all disagreements were discussed until a consensus was reached. The results of this phase have a Cohen's kappa of 0.822, showing a substantial-level of agreement [57].

4.2.1 Evaluation Metric. If the caption pair $p = (I(b), I(i'))$ is reported as suspicious issue, and $I(b)$ or/and $I(i')$ contains captioning error(s), then we set $error(p)$ to be true, otherwise we set $error(p)$ to be false. Given a list of suspicious issues, the precision

[2] https://stanfordnlp.github.io/stanza/

Table 1: Precision (true positives/suspicious issues) of Deeptest [73] and MetaIC on one paid API and five models.

IC Systems	Deeptest (Perturbation Method)				MetaIC (Overlapping Ratio)			
	Blur	Brightness	Contrast	Shear	0%	15%	30%	45%
Attention	35.0% (185/528)	37.9% (120/317)	37.2% (196/527)	41.1% (245/596)	98.0% (948/967)	97.7% (937/959)	98.4% (948/963)	98.2% (948/965)
Oscar$_B$	20.2% (127/630)	14.7% (38/258)	18.4% (96/521)	21.4% (119/555)	91.3% (652/714)	91.4% (637/697)	91.2% (667/731)	92.2% (694/753)
Oscar$_L$	19.8% (121/610)	12.9% (36/279)	17.4% (91/522)	18.5% (100/542)	92.3% (624/676)	91.7% (620/676)	91.2% (625/685)	91.6% (647/706)
VinVL$_B$	34.2% (207/606)	26.2% (113/431)	29.1% (167/574)	28.6% (185/646)	88.0% (563/640)	87.3% (552/632)	88.4% (571/646)	88.5% (598/676)
VinVL$_L$	21.8% (131/602)	16.5% (60/363)	16.9% (98/579)	19.3% (113/586)	86.7% (535/617)	86.0% (535/622)	84.9% (535/630)	85.1% (560/658)
Microsoft Azure API	39.8% (181/455)	41.2% (56/136)	41.2% (163/396)	38.6% (197/511)	96.6% (858/888)	96.1% (852/887)	96.5% (859/890)	97.4% (860/883)

is calculated by:

$$Precision = \frac{\sum_{p \in P} error(p)}{|P|}, \tag{9}$$

where P are the suspicious issues reported by MetaIC and $|P|$ is the number of the suspicious issues.

4.2.2 Results. The results are presented in Table 1, the precision of MetaIC and Deeptest [73] on one API and five models. The precision of MetaIC ranges from 84.9% to 98.4%, while Deeptest's precision ranges from 12.9% to 41.2%. Specifically, MetaIC achieves a precision of (96.1%-97.4%) on Microsoft Azure API, and a precision of (84.9%-98.4%) on the five IC models. The results demonstrate that MetaIC is much more precise in testing IC systems than existing metamorphic testing techniques for systems that also take images as input. In addition, MetaIC reports much more erroneous issues. Specifically, MetaIC reports 535 to 948 erroneous issues, while Deeptest reports 36 to 245. MetaIC successfully report 16,825 erroneous issues, which contains a total number of 17,380 captioning errors. Specifically, out of the 16,825 erroneous issues reported, 120 issues are reported because of captioning errors in the original images. Note that the comparison between MetaIC and Deeptest is not apple-to-apple as Deeptest was originally designed for testing image classifiers. We regards Deeptest as a baseline here for the completeness of discussion. The images synthesized by MetaIC are diverse. From the table, we can observe that MetaIC consistently achieves high precision under different overlapping ratios, showing the effectiveness of our object resizing and location tuning algorithms. For example, when we use MetaIC to test the Microsoft Azure API [5], the precision values for $\{ratio_0, ratio_1, ratio_2, ratio_3\}$ are $\{96.6\%, 96.1\%, 96.5\%, 97.4\%\}$, respectively.

4.2.3 False positives. To further understand the results, we categorize the false positives of MetaIC and discuss the prospective solutions for them. Although MetaIC achieves very high precision in our evaluation, we still encounter some false positives. We manually inspect the false positives and present three typical examples in Fig. 7. First, after inserting a bird into the background image, the caption describes the "bird" as "parrot", which is correct because parrots are subspecies of birds. However, our class set does not contain "parrot", leading to a false positive. Second, after inserting a cow into the background image, the depiction of the person changes from "person" to "woman". A dictionary containing more object types can effectively reduce these two kinds of false positives. Third, "a pair of scissors" is regarded as an incorrect singular-plural form of "scissors" by our method, while in modern English, the word "scissors" has no singular form, leading to the false positive. In the future, adopting a more advanced POS tagging tool or maintaining a dictionary of words of special singular-plural form can help.

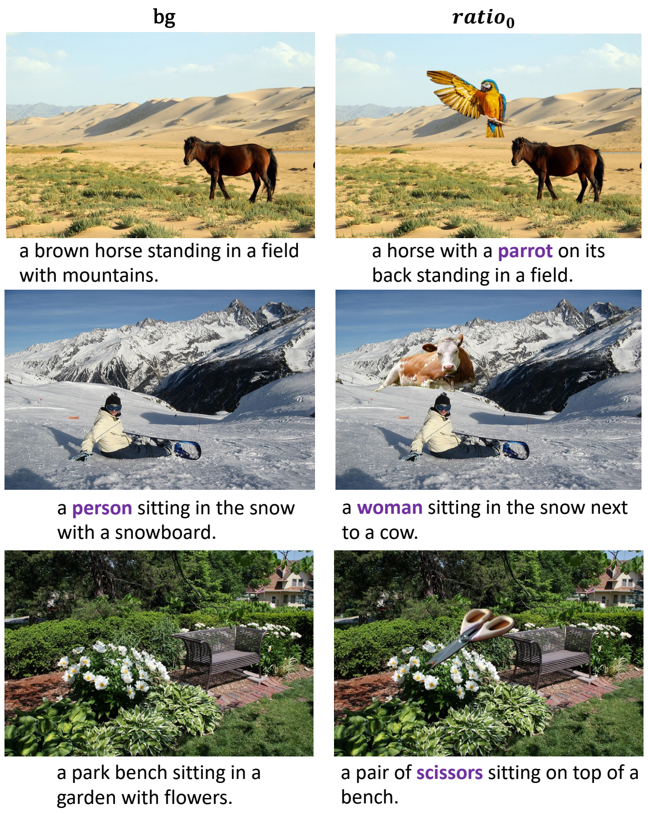
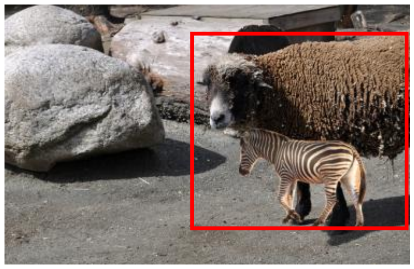

bg
$ratio_0$
a brown horse standing in a field with mountains.
a horse with a parrot on its back standing in a field.
a person sitting in the snow with a snowboard.
a woman sitting in the snow next to a cow.
a park bench sitting in a garden with flowers.
a pair of scissors sitting on top of a bench.

Figure 7: False positives reported by MetaIC.

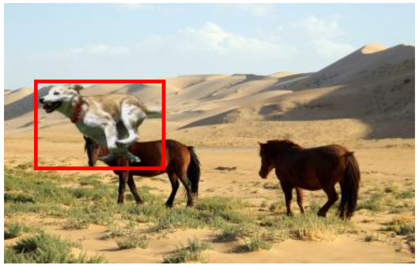
4.3 Erroneous Captions

MetaIC is capable of finding three kinds of captioning errors:

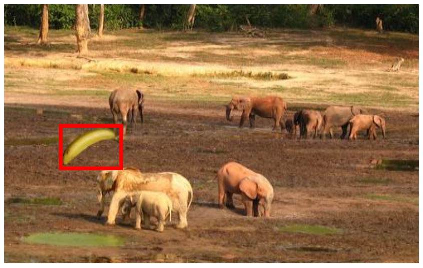
- **Classification error** is an error that IC systems provide an incorrect class name for an object, for instance, an erroneous caption depicts an airplane as a skateboard.
- **Recognition error** is an error that IC systems omit the description of a salient object in the image.
- **Single-plural error** is an error that IC systems return the incorrect singular-plural form of an noun in the caption. For



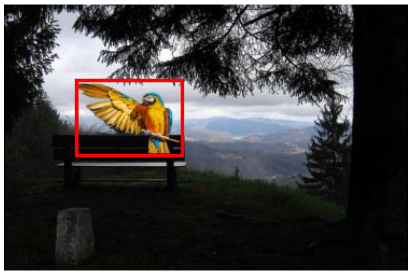
a couple of **zebra** standing next to each other. (*misclassify the **sheep** as the **zebra***)



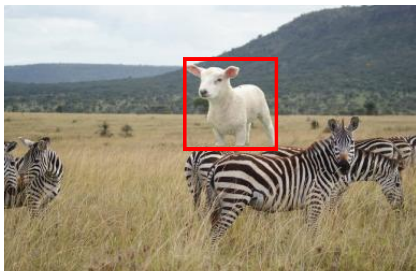
a group of horses stand in a grassy field (*omission of the **dog***)



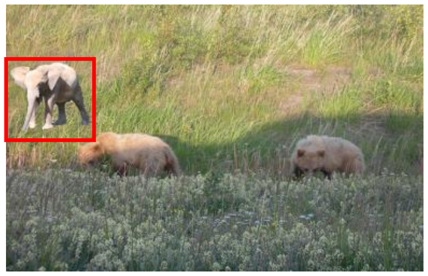
a herd of elephants standing in a field with **bananas** (*wrong singular-plural form of the **banana***)



a **person** in a garment sitting on a bench (*misclassify the **bird** as the **person***)



a herd of zebra standing on top of a lush green field. (*omission of the **sheep***)



a group of **elephants** and two bears in a field (*wrong singular-plural form of the **elephant***)

Figure 8: Examples of captioning errors reported by MetaIC.

Table 2: Ablation study of the metamorphic relations.

IC Systems	Overlapping Ratio							
	0%		15%		30%		45%	
	MR_1	MR_1+MR_2	MR_1	MR_1+MR_2	MR_1	MR_1+MR_2	MR_1	MR_1+MR_2
Attention	98.1% (935/953)	98.0% (948/967)	97.8% (924/945)	97.7% (937/959)	98.5% (932/946)	98.4% (948/963)	98.2% (936/953)	98.2% (948/965)
Oscar_B	92.5% (626/677)	91.3% (652/714)	92.5% (604/653)	91.4% (637/697)	92.6% (636/687)	91.2% (667/731)	93.5% (659/705)	92.2% (694/753)
Oscar_L	93.4% (606/649)	92.3% (624/676)	92.4% (595/644)	91.7% (620/676)	91.8% (592/645)	91.2% (625/685)	92.0% (619/673)	91.6% (647/706)
VinVL_B	89.9% (543/604)	88.0% (563/640)	88.9% (526/592)	87.3% (552/632)	90.2% (540/599)	88.4% (571/646)	90.0% (568/631)	88.5% (598/676)
VinVL_L	88.1% (518/588)	86.7% (535/617)	87.7% (519/592)	86.0% (535/622)	86.9% (517/595)	84.9% (535/630)	87.2% (540/619)	85.1% (560/658)
Microsoft Azure API	96.7% (844/873)	96.6% (858/888)	96.3% (833/865)	96.1% (852/887)	96.7% (842/871)	96.5% (859/890)	97.4% (839/861)	97.4% (860/883)

example, an erroneous caption gives a depiction "bananas" when there is only one banana in the image.

To provide a glimpse of the diversity of the uncovered errors, this section highlights examples for all the three kinds of errors in Fig. 8. The first column corresponds to the classification errors, *e.g.*, the caption in the first row uses "a couple of zebra" to describe the image including a sheep and a zebra, and the caption of the second row incorrectly describes the bird as a person in a garment. The second column corresponds to the recognition errors, where the caption of the first row misses the depiction of a dog near two horses, and the caption of the second row omits the sheep in a group of zebras. The third column shows the errors of singular-plural form, where the caption of the first row describes a single banana as bananas, and the caption of the second row depicts a single elephant as two elephants.

4.4 Ablation Study

We conduct the ablation study to evaluate the effectiveness of the two metamorphic relations we propose. From Table 2 we can see that the precision of MR_1 and $MR_1 + MR_2$ is very close to each other in most cases. Sometimes, the precision will be slightly decreased when we use $MR_1 + MR_2$, while we will report more erroneous issues. For example, for the $Oscar_B$ model with 45% overlapping ratio, precision is decreased from 93.5% to 92.2%. However, with $MR_1 + MR_2$, we can find 35 more erroneous issues than only using MR_1, which shows that MR_2 is useful and necessary.

4.5 Case Study on IC Errors via Visualization

To better understand the reported captioning errors and explore the potential root causes, in this section, we conduct visualization experiments. Specifically, we try to answer the question: "why the IC systems return incorrect captions given the synthesized images?" We visualize the errors in both CNN-RNN-Based IC systems (via attention of the generated object words) and VLP IC systems (via the prediction of faster R-CNN).

4.5.1 *CNN-RNN-Based IC.*

For CNN-RNN-Based IC, we explore errors in Attention [85] because it achieves high accuracy among all the CNN-RNN-Based ICs. We visualize the attention mechanism as it is the most important component of the model. Specifically, we intend to show how different overlapping ratios would affect the attention mechanism and the caption.

In Fig. 9, we present the attention mask changes after inserting an object to the original background image. We could observe that, at the beginning, the black bear is correctly captioned in the background image. However, when we insert the horse into the background image with different overlapping ratios ($n = 4$, $ratio_{max} = 45\%$), we find that the captions become incorrect. For $ratio_0$ and $ratio_1$, the black horse is misclassified as a black bear. For $ratio_2$, the black horse and black bear have been both recognized as black sheep. For $ratio_3$, the black bear is misclassified as black horse. The second row of Fig. 9 shows the attention masks generated by the model, which serves as an important component in its captioning process. By observing the attention masks of $ratio_0$ and $ratio_1$, we can see that the attention masks for the two words of "bear" in the caption mistakenly highlight the regions of both the horse and the bear. This indicates that the captioning error is likely to be caused by model's attending to incorrect region. Thus, research on improving the attention mechanism in demand.

4.5.2 *Perturbation to VLP IC.*

MetaIC also reports many erroneous captions produced by VLP ICs. Specifically, we choose a typical VLP IC model $Oscar_B$ and conduct visualization experiment to show the vulnerabilities of the two components of VLP ICs, including Faster R-CNN in the first stage and BERT in the second stage. By comparing Fig. 10 (a) with Fig. 10 (b), we show an example where its erroneous caption is caused by Faster R-CNN (the first component). We draw the bounding boxes produced by the Faster R-CNN of $Oscar_B$ for the background and the synthesized image. For clarity, we only draw the bounding boxes for the object in the image. The caption produced by $Oscar_B$ is correct for the background image. However, for the synthesized image, the caption describes a single giraffe as "a couple of giraffes", resulting in an error of singular-plural form. We could observe that the bounding boxes for background image are of high accuracy. However, the bounding box of the giraffe in the synthesized image and that of the horses have a big overlap. The part of the image framed by the bounding box will be encoded into region-feature, which acts as the major input to the second component of $Oscar_B$. The second component uses region features and the corresponding tags (*e.g.*, "giraffe") to generate the caption. Specifically, $Oscar_B$ concatenates the tag "giraffe" with the region-feature for cross-modal representation learning. We think the erroneous depiction of "a couple of giraffes" could be caused by the low-quality region-feature of "giraffe" (*i.e.*, the sub-optimal bounding box). It overlaps a lot with the horses, and thus $Oscar_B$ captions one of the horses as a giraffe.

The visualization experiment also shows that the erroneous captions may be caused by the glitches in the BERT module the second component of VLP ICs. As shown in Fig. 10 (c), the bounding boxes of the objects in the image are of high accuracy, while the generated sentence omits the description of the bowl on the frisbee. The visualization case study indicates that MetaIC can report captioning errors cause by both the first component (image) and the second component (text).

4.6 Finding Labeling Errors in the Training Corpus

Labeling errors in the training corpus has been a prevalent and severe problem in machine learning and deep learning. Even the most famous Datasets have errors in their labels, such as ImageNet [43], CIFAR [42], and MNIST [23]. Several methods [65, 88] have been proposed to tackle this problem, mainly focusing on noises in the data. Recently, AI software applications are approaching human-level performance. For IC task, Hu *et al.* [37] claims that their model has achieved a better result than human in terms of CIDEr [77] score in Nocaps [8] task. Given that these models have been well trained on the existing corpora, fixing labeling errors in these corpora is a reasonable way to further improve their performance. In this section, we explore whether it is possible to adapt the high-level idea of MetaIC detecting labelling errors in a standard dataset MS COCO Caption [21].

Specifically, denote $GT(b)$ as the label caption of image b in MS COCO Caption, we have a tuple $(I(b), I(i'), GT(b))$. Denote the common object class set between $I(b)$ and $I(i')$ as Set_{invar}:

$$Set_{invar} = Set(I(b)) \cap Set(I(i')), \tag{10}$$

Intuitively, objects classes in both $(I(b)$ and $I(i')$ imply the important objects in the image. We denote the object class set of $GT(b)$ as $Set(GT(b))$, it should satisfy the relation that:

$$Set_{invar} \subset Set(GT(b)). \tag{11}$$

Otherwise, $GT(b)$ may depict image b improperly since it is likely to miss important objects in its caption. We map the object class to their super-category defined in MS COCO Caption and use the super-category in the aforementioned method.

We choose ten object classes $\{dog, cat, sheep, truck, cow, zebra, elephant, horse, frisbee, bird\}$, construct 6,662 tuples of $(I(b), I(i'), GT(b))$, and try to find erroneous labels of these classes in MS COCO Caption. By using this technique adapted from MetaIC, we have found 151 caption errors, using only a small portion of the data in MS COCO Caption. We present examples of incorrect label errors in Fig. 11, including typos, misclassification error, "no-image" error, *etc.* For example, caption (a) in the first row mistakenly writes "at" rather than "cat"; caption (c) mistakenly describe a "frisbee" as a "pizza"; caption (e) is a complain message rather than a caption; and caption (d) incorrectly depicts a "dog" as a "man".

Given that the performance of deep learning models mainly depends on the model structure and the parameters learned from the training dataset, we believe the labeling errors we find with MetaIC indicate the usefulness and wide applicability in enhancing the robustness of IC software. Thus, it reflects the practical relevance of MetaIC. The 151 labeling errors are found from 6,662 captioned images in the training data. We believe our approach can detect more errors on a larger input set. Although finding labeling errors is not the main focus of this paper, we think the experiment demonstrates the wide-applicability of MetaIC.

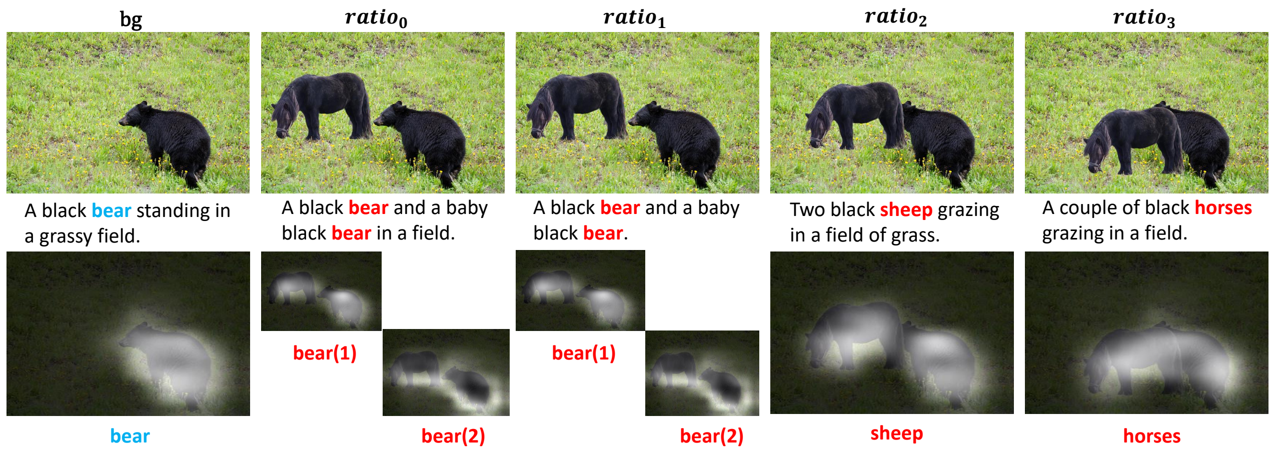
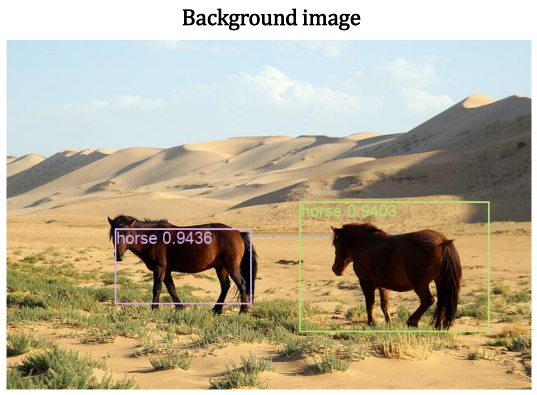
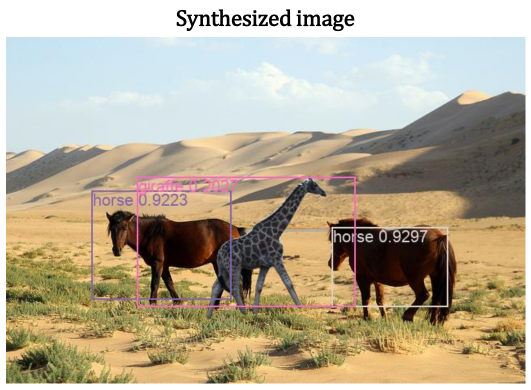

bg
ratio0
ratio1
ratio2
ratio3
A black bear standing in a grassy field.
A black bear and a baby black bear in a field.
A black bear and a baby black bear.
Two black sheep grazing in a field of grass.
A couple of black horses grazing in a field.
bear
bear(1)
bear(2)
bear(1)
bear(2)
sheep
horses

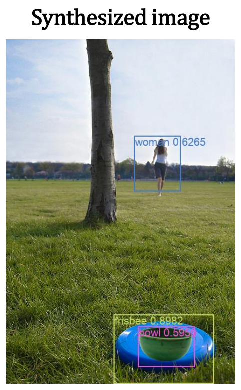
Figure 9: Attention mechanism fails to depict the images.



(a) a couple of horses standing in the grass in a field.



(b) a couple of giraffes and a horse walking in a field.



(c) a woman standing in a field with a blue frisbee.

Figure 10: Errors caused by the first component (a)(b) and the second component (c) in VLP IC.

4.7 Retraining with Erroneous Issues

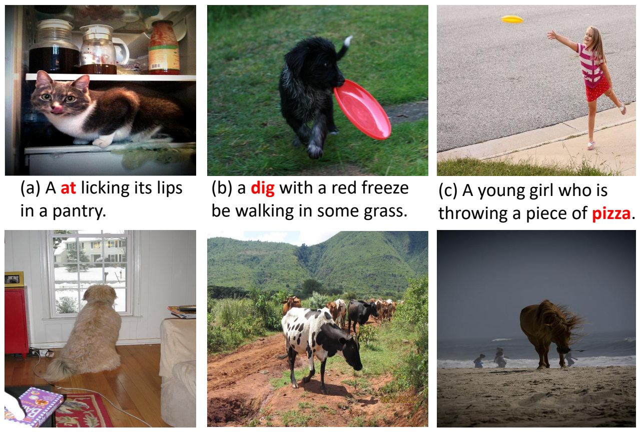
To explore whether the erroneous issues reported by MetaIC can be utilized to improve IC systems, we re-label the erroneous issues following the labeling standard of MS COCO Caption [21], and fine-tune the $Oscar_B$ model. Specifically, we add the re-labeled synthesized images to MS COCO Caption and perform fine-tuning for 40K steps. We follow the publicly available splits[3] as in the original Oscar paper [47] and use our synthesized images as the augmented data in the training set.

Table 3 presents the results. We can observe that BLEU-4 increases by 1.2%, METEOR increases by 3.7%, CIDEr increases by 2.5%, and SPICE increases by 8.8%. Specifically, the SPICE [9] has a system-level correlation of 0.88 with human judgements on MS COCO [51]. According to a survey of IC systems (Table 2 in [69]), our improvement of the four score are significant. In addition, we can see that except for BLEU-4, the other scores of $Oscar_{B-Fintune}$ are higher than $Oscar_L$. The fine-tuned model achieves competitive performance with much fewer parameters and less training time. The results indicate that the captioning errors in the synthetic images can be utilized to further improve the performance of IC systems, both in efficiency and accuracy, demonstrating its practical relevance. Note that although improving IC system using test cases is an interesting and important topic, it is not the focus of this paper. Thus, we regard it as a promising future direction.

Table 3: Performance of fine-tuned model.

Model	Evaluation Metric			
	BLEU-4	METEOR	CIDEr	SPICE
$Oscar_B$	40.5	29.7	137.6	22.8
$Oscar_L$	41.7	30.6	140.0	24.5
$Oscar_{B\text{-}Finetune}$	41.0	30.8	141.1	24.8
Improvement (%)	1.2	3.7	2.5	8.8

[3]https://cs.stanford.edu/people/karpathy/deepimagesent/



(a) A at licking its lips in a pantry.
(b) a dig with a red freeze be walking in some grass.
(c) A young girl who is throwing a piece of pizza.
(d) a man that is sitting by a window staring out the window
(e) There is no image here to provide a caption for.
(f) a camel walking on a beach towards the water

Figure 11: Labeling errors in MS COCO Caption [21] reported by MetaIC.

5 RELATED WORK

5.1 Robust AI Software

In recent years, deep neural networks have achieved great success in a variety of fields and thus artificial intelligence (AI) software has been widely used in our daily lives. However, AI software can generate erroneous outputs that cause severe accidents and even endanger the users [44, 46, 93]. To explore the vulnerability of AI software, a line of research has focused on attacking various AI systems, such as classification systems [28, 52, 91], and automatic speech recognition systems [14, 62]. These papers fool the AI software with imperceptible perturbations. Meanwhile, to enhance the reliability of AI software, a variety of related topics have been studied, including testing [25, 26, 33–35, 39], online adversarial detection [29, 53, 72, 81], and robust training mechanisms for deep neural networks [40, 50, 54, 61]. Most of these methods are white-box, largely dependent on the knowledge of the model internals; while our approach is black-box, which can be easily adapted to test any IC systems.

5.2 Multimodal Task and Image Captioning

Modality refers to the way in which something happens or being sensed by human. Accordingly, a multimodal task indicates that the task involves more than one modality (*e.g.*, image and text for image captioning). The corresponding AI models need to learn these multimodal signals collectively, which brings new challenges as introduced by Baltrušaitis et al. [11]: representation, translation, alignment, fusion, and co-learning. Typical multimodal tasks include speech recognition and synthesis [30, 38], cross-modal retreival [27, 80], and image captioning [47, 78, 85, 90]. Compared with single-modal systems (*e.g.*, image classifier or machine translation), multimodal systems are more difficult to test because we need to consider the translation from one modality to another and most of the existing testing techniques for single-modal models cannot be used here without non-trivial adaptations.

Recent years have witnessed the blossom of IC systems, where researchers focus on improving the accuracy of IC. For CNN-RNN-Based IC, researchers designed various CNNs and RNNs to enhance the intermediate representation [41, 78, 85]. For VLP IC, researchers proposed multiple pre-trained techniques [37, 47, 90, 92]. Different from these papers that make effort to achieve high accuracy, MetaIC aims to improve the robustness of IC. To this end, several papers focus on attacking existing IC models in a white-box manner [18, 86, 87], which requires the complete knowledge of underlying networks. Differently, MetaIC is black-box, which does not rely on model internals, such as network structure and parameters.

5.3 Metamorphic Testing

Metamorphic Testing (MT) is a general methodology that applies a transformation to test input(s) and observes how the program output turns into a different one as a result [19, 20, 67]. The core idea of MT is to verify the MRs between the outputs from multiple runs of the program with different inputs, which is useful when test oracle is lacking. MT is widely adopted in traditional software, such as compilers [45, 49], datalog engines [55], and service-oriented applications [16, 17]. Recently, it has also been used in testing AI software, such as autonomous cars [73, 89], statistical classifiers [83, 84], object detection [82], and machine translation [33, 34]. In this paper, we propose MetaIC, a novel, widely-applicable metamorphic testing approach for image captioning.

6 CONCLUSION

In this paper, we propose the first black-box testing approach, MetaIC, for validating image captioning systems. The distinct benefits of MetaIC are its simplicity and generality, and thus wide applicability. MetaIC can effectively disclose many captioning errors. In our experiments, MetaIC successfully reports 16,825 erroneous issues in six IC systems with high precision (84.9%-98.4%) revealing 17,380 captioning errors. In addition to the main focus of this paper (*i.e.*, testing), to further understand the reported errors, we visualize the attention of objects in CNN-RNN-Based ICs and the predictions of Faster R-CNN used in VLP ICs, which explores the major root causes behind. We also show that MetaIC can be adapted to find errors in the standard dataset of image captioning, indicating its potential in further enhancing the reliability of IC systems. For future work, we plan to extend MetaIC by considering other kinds of caption constituents (*e.g.*, verbs). We will also explore automated error detection for standard datasets, which we regard as an important future direction.

ACKNOWLEDGMENTS

We thank the anonymous ISSTA reviewers for their valuable feedback on the earlier draft of this paper. This paper was supported by the National Natural Science Foundation of China (No. 62102340). Part of the experiments were conducted on the high-performance computing platform, which is managed by the Information Technology Services Office (ITSO) at the Chinese University of Hong Kong, Shenzhen.

REFERENCES

[1] 2018. *Image Caption Generator.* https://developer.ibm.com/exchanges/models/all/max-image-caption-generator

[2] 2020. *Auto Image Captioning.* https://medium.com/ai-techsystems/auto-image-captioning-8efcfa517402

[3] 2020. *Automatic Image Captioning Using Neural Networks.* https://evergreen.team/articles/automatic-image-captioning.html

[4] 2021. *Automated Image Captions and Descriptions.* https://cloud.google.com/ai-workshop/experiments/automated-image-captions-and-descriptions

[5] 2021. *Azure Cognitive Services.* https://azure.microsoft.com/en-us/services/cognitive-services

[6] 2021. *Flickr: Find your inspiration.* https://www.flickr.com/

[7] 2022. *MetaIC: An Automated Testing Toolkit for Image Captioning.* https://github.com/RobustNLP/TestIC

[8] Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. nocaps: novel object captioning at scale. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR).* 8948–8957.

[9] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation (ECCV). In *European conference on computer vision.* Springer, 382–398.

[10] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[11] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)* 41, 2 (2018), 423–443.

[12] D Bolya, C Zhou, F Xiao, and YJ Lee. 2020. YOLACT++: Better Real-time Instance Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2020).

[13] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. 2019. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR).* 9157–9166.

[14] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW).* IEEE, 1–7.

[15] Mariano Ceccato, Massimiliano Di Penta, Paolo Falcarin, Filippo Ricca, Marco Torchiano, and Paolo Tonella. 2014. A family of experiments to assess the effectiveness and efficiency of source code obfuscation techniques. *Empirical Software Engineering (ESE)* 19, 4 (2014), 1040–1074.

[16] WK Chan, Shing Chi Cheung, and Karl RPH Leung. 2005. Towards a metamorphic testing methodology for service-oriented software applications. In *Fifth International Conference on Quality Software (QSIC'05).* 470–476.

[17] Wing Kwong Chan, Shing Chi Cheung, and Karl RPH Leung. 2007. A metamorphic testing approach for online testing of service-oriented software applications. *International Journal of Web Services Research (IJWSR)* 4, 2 (2007), 61–81.

[18] Hongge Chen, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, and Cho-Jui Hsieh. 2018. Attacking Visual Language Grounding with Adversarial Examples: A Case Study on Neural Image Captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers).* 2587–2597.

[19] Tsong Y Chen, Shing C Cheung, and Shiu Ming Yiu. 2020. Metamorphic testing: a new approach for generating next test cases. *arXiv preprint arXiv:2002.12543* (2020).

[20] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, TH Tse, and Zhi Quan Zhou. 2018. Metamorphic testing: A review of challenges and opportunities. *ACM Computing Surveys (CSUR)* 51, 1 (2018), 1–27.

[21] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325* (2015).

[22] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[23] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine (IEEE Signal Process Mag)* 29, 6 (2012), 141–142.

[24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[25] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. 2019. Deepstellar: Model-based quantitative analysis of stateful deep learning systems. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE).* 477–487.

[26] Alessio Gambi, Marc Mueller, and Gordon Fraser. 2019. Automatically testing self-driving cars with search-based procedural content generation. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA).* 318–328.

[27] Dehong Gao, Linbo Jin, Ben Chen, Minghui Qiu, Peng Li, Yi Wei, Yi Hu, and Hao Wang. 2020. Fashionbert: Text and image matching with adaptive loss for cross-modal retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).* 2251–2260.

[28] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[29] Shuangchi Gu, Ping Yi, Ting Zhu, Yao Yao, and Wei Wang. 2019. Detecting adversarial examples in deep neural networks using normalizing filters. *UMBC Student Collection* (2019).

[30] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567* (2014).

[31] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision (CVPR).* 2961–2969.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR).* 770–778.

[33] Pinjia He, Clara Meister, and Zhendong Su. 2020. Structure-invariant testing for machine translation. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE).* 961–973.

[34] Pinjia He, Clara Meister, and Zhendong Su. 2021. Testing Machine Translation via Referential Transparency. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE).* 410–422.

[35] Jens Henriksson, Christian Berger, Markus Borg, Lars Tornberg, Cristofer Englund, Sankar Raman Sathyamoorthy, and Stig Ursing. 2019. Towards structured evaluation of deep neural network supervisors. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest).* 27–34.

[36] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[37] Xiaowei Hu, Xi Yin, Kevin Lin, Lijuan Wang, Lei Zhang, Jianfeng Gao, and Zicheng Liu. 2020. Vivo: Surpassing human performance in novel object captioning with visual vocabulary pre-training. *arXiv preprint arXiv:2009.13682* (2020).

[38] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron Oord, Sander Dieleman, and Koray Kavukcuoglu. 2018. Efficient neural audio synthesis. In *International Conference on Machine Learning (ICML).* 2410–2419.

[39] Sungmin Kang, Robert Feldt, and Shin Yoo. 2020. SINVAD: Search-based Image Space Navigation for DNN Image Classifier Test Input Generation. In *Proceedings of the International Workshop on Search Based Software Testing (SBST 2020).*

[40] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. 2018. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373* (2018).

[41] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR).* 3128–3137.

[42] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[43] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems (NIPS)* 25 (2012), 1097–1105.

[44] Fred. Lambert. 2016. *Understanding the fatal Tesla accident on Autopilot and the NHTSA probe.* https://electrek.co/2016/07/01/understanding-fatal-tesla-accident-autopilot-nhtsa-probe/

[45] Vu Le, Mehrdad Afshari, and Zhendong Su. 2014. Compiler validation via equivalence modulo inputs. *ACM Sigplan Notices* 49, 6 (2014), 216–226.

[46] Sam Levin. 2018. *Tesla fatal crash: 'autopilot' mode sped up car before driver killed, report finds.* https://www.theguardian.com/technology/2018/jun/07/tesla-fatal-crash-silicon-valley-autopilot-mode-report

[47] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, and Furu Wei. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision (ECCV).* Springer, 121–137.

[48] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. 2017. Fully convolutional instance-aware semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR).* 2359–2367.

[49] Christopher Lidbury, Andrei Lascu, Nathan Chong, and Alastair F Donaldson. 2015. Many-core compiler fuzzing. *ACM SIGPLAN Notices* 50, 6 (2015), 65–76.

[50] Ji Lin, Chuang Gan, and Song Han. 2019. Defensive quantization: When efficiency meets robustness. *arXiv preprint arXiv:1904.08444* (2019).

[51] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV).* Springer, 740–755.

[52] Bo Luo, Yannan Liu, Lingxiao Wei, and Qiang Xu. 2018. Towards imperceptible and robust adversarial example attacks against neural networks. In *Thirty-second*

aaai conference on artificial intelligence (AAAI).
[53] Shiqing Ma and Yingqi Liu. 2019. Nic: Detecting adversarial samples with neural network invariant checking. In *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS 2019)*.
[54] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).
[55] Muhammad Numair Mansur, Maria Christakis, and Valentin Wüstholz. 2021. Metamorphic testing of Datalog engines. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (FSE/ESEC)*. 639–650.
[56] Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. (1993).
[57] Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica* 22, 3 (2012), 276–282.
[58] Youssef Mroueh. 2020. *Image Captioning as an Assistive Technology*. https://www.ibm.com/blogs/research/2020/07/image-captioning-assistive-technology
[59] CAROL VAN NATTA. 2020. AI Fails at Photo Captions. https://author.carolvannatta.com/ai-fails-at-photo-captions/
[60] World Health Organization. 2019. World report on vision. (2019). ISBN: 9241516577 Publisher: World Health Organization.
[61] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*. 582–597.
[62] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. 2019. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International conference on machine learning*. PMLR, 5231–5240.
[63] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems (NIPS)* 28 (2015), 91–99.
[64] Andrea Romdhana, Mariano Ceccato, Gabriel Claudiu Georgiu, Alessio Merlo, and Paolo Tonella. 2021. COSMO: Code Coverage Made Easier for Android. In *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 417–423.
[65] Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M Aroyo. 2021. "Everyone wants to do the model work, not the data work": Data Cascades in High-Stakes AI. In *proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI)*. 1–15.
[66] Stan Schroeder. 2016. Microsoft created a bot to auto-caption photos and it's going hilariously wrong. https://mashable.com/article/microsoft-captionbot Section: Life.
[67] Sergio Segura, Gordon Fraser, Ana B Sanchez, and Antonio Ruiz-Cortés. 2016. A survey on metamorphic testing. *IEEE Transactions on software engineering (TSE)* 42, 9 (2016), 805–824.
[68] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229* (2013).
[69] Matteo Stefanini, Marcella Cornia, Lorenzo Baraldi, Silvia Cascianelli, Giuseppe Fiameni, and Rita Cucchiara. 2021. From show to tell: A survey on image captioning. *arXiv preprint arXiv:2107.06912* (2021).
[70] Zeyu Sun, Jie M Zhang, Mark Harman, Mike Papadakis, and Lu Zhang. 2020. Automatic testing and improvement of machine translation. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE)*. 974–985.
[71] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems (NIPS)*. 3104–3112.
[72] Guanhong Tao, Shiqing Ma, Yingqi Liu, and Xiangyu Zhang. 2018. Attacks meet interpretability: Attribute-steered detection of adversarial samples. *arXiv preprint arXiv:1810.11580* (2018).
[73] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering (ICSE)*. 303–314.
[74] Yuchi Tian, Ziyuan Zhong, Vicente Ordonez, Gail Kaiser, and Baishakhi Ray. 2020. Testing DNN image classifiers for confusion & bias errors. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE)*. 1122–1134.
[75] Jonas Uhrig, Eike Rehder, Björn Fröhlich, Uwe Franke, and Thomas Brox. 2018. Box2pix: Single-shot instance segmentation by assigning pixels to object boxes. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. 292–299.
[76] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems (NIPS)*. 5998–6008.
[77] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 4566–4575.
[78] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 3156–3164.
[79] Alessio Viticchié, Leonardo Regano, Marco Torchiano, Cataldo Basile, Mariano Ceccato, Paolo Tonella, and Roberto Tiella. 2016. Assessment of source code obfuscation techniques. In *2016 IEEE 16th international working conference on source code analysis and manipulation (SCAM)*. IEEE, 11–20.
[80] Nam Vo, Lu Jiang, Chen Sun, Kevin Murphy, Li-Jia Li, Li Fei-Fei, and James Hays. 2019. Composing text and image for image retrieval-an empirical odyssey. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6439–6448.
[81] Jingyi Wang, Guoliang Dong, Jun Sun, Xinyu Wang, and Peixin Zhang. 2019. Adversarial sample detection for deep neural network through model mutation testing. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. 1245–1256.
[82] Shuai Wang and Zhendong Su. 2020. Metamorphic object insertion for testing object detection systems. In *2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. 1053–1065.
[83] Xiaoyuan Xie, Joshua Ho, Christian Murphy, Gail Kaiser, Baowen Xu, and Tsong Yueh Chen. 2009. Application of metamorphic testing to supervised classifiers. In *2009 Ninth International Conference on Quality Software (QSIC)*. 135–144.
[84] Xiaoyuan Xie, Joshua WK Ho, Christian Murphy, Gail Kaiser, Baowen Xu, and Tsong Yueh Chen. 2011. Testing and validating machine learning classifiers by metamorphic testing. *Journal of Systems and Software (JSS)* 84, 4 (2011), 544–558.
[85] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning (ICML)*. PMLR, 2048–2057.
[86] Xiaojun Xu, Xinyun Chen, Chang Liu, Anna Rohrbach, Trevor Darrell, and Dawn Song. 2018. Fooling vision and language models despite localization and attention mechanism. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4951–4961.
[87] Yan Xu, Baoyuan Wu, Fumin Shen, Yanbo Fan, Yong Zhang, Heng Tao Shen, and Wei Liu. 2019. Exact adversarial attack to image captioning via structured output learning with latent variables. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4135–4144.
[88] Jing Zhang, Victor S Sheng, Tao Li, and Xindong Wu. 2017. Improving crowd-sourced label quality using noise correction. *IEEE transactions on neural networks and learning systems (TNNLS)* 29, 5 (2017), 1675–1688.
[89] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 2018. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. 132–142.
[90] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021. VinVL: Making Visual Representations Matter in Vision-Language Models. *arXiv preprint arXiv:2101.00529* (2021).
[91] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. 2020. Towards large yet imperceptible adversarial image perturbations with perceptual color distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1039–1048.
[92] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 34. 13041–13049.
[93] Chris. Ziegler. 2016. *A Google self-driving car caused a crash for the first time*. https://www.theverge.com/2016/2/29/11134344/google-self-driving-car-crash-report